

Change Report

Group 18

Team B

Olivia Betts
Zac Bhumgara
Nursyarmila Ahmad Shukri
Cameron Duncan-Johal
Muaz Waqas
Oliver Northwood
Teddy Seddon

The main tool we used to track changes was the Google Docs version history, and the Git blame system. Google Docs version history was used to track changes with the written deliverables, as that's where they were all written and shared; Git blame was used to track changes with the website and game code repos.

Other, more specific tools we used also had version histories. For example, LucidChart (used to make the architectural diagrams) gave us the ability to see previous versions of the diagrams.

All of these showed how the files changed from their originals to the new files, as well as who made the changes and when. We also had the ability to roll back changes and return to old versions of each file, if anything went wrong.

Requirements

Original pdf: <https://imozwastaken.github.io/pdf/Req1.pdf>

Updated pdf:

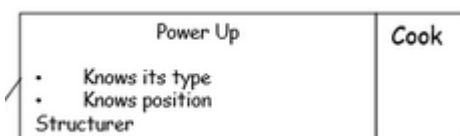
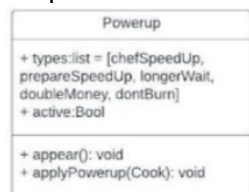
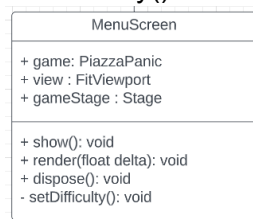
Change made	Justification
New functional requirements based on existing user requirements	<p>We added functional requirements to aid our implementation of the existing user requirements. These were:</p> <ul style="list-style-type: none">- FR_CONFIGURE_CUSTOMERS and FR_CUSTOMER_ARRIVALS (indices 14.1 and 14.2, respectively), discussing the amount of customers visiting in scenario mode, and the rate of customer arrivals during endless mode- FR_AVOID_RESETS (index 4.4), a sub-requirement of UR_STATION_ACTION, ensuring that interacting with a station or moving to a different screen doesn't reset the cooks' or customers' locations
New user requirements (and the corresponding functional requirements)	<p>New requirements added were:</p> <ul style="list-style-type: none">- Support for powerups. This was given the ID UR_POWERUPS (index 15, and will be referred to later in the document as Req15) in the user requirements table. From here, we identified the key function requirements needed to accomplish it - FR_POWERUP_EFFECT and FR_POWERUP_APPEARANCE (indices 15.1 and 15.2 in the functional requirements table)- The ability to select a difficulty to play on, given the ID UR_DIFFICULTY (index 16, to be called Req16). This had an associated functional requirement of FR_SELECT_DIFFICULTY, with an of index 16.1- The ability to save and load games, in the user requirements table with an ID of UR_SAVESTATES (index 17, Req17). Another two functional requirements were associated with this - FR_SAVE_GAME and FR_LOAD_GAME, with indices of 17.1 and 17.2 respectively- Ensuring all code is readable and understandable - referred to UR_READABLE_CODE and Req19 <p>We contacted the customer again to confirm that these requirements were correct and complete. This led us to add the additional requirement missed by the first group - that all assets and technologies should be appropriately licensed. This was added to the user requirements table as UR_LICENCE, at index 18. No functional requirements were associated with this one.</p>
New non-functional requirements	<p>To existing requirement UR_USER_EXPERIENCE, we added the requirement NEF_WINDOW_RESIZE, based off user feedback</p> <p>For Req19, we added the following requirements, to ensure we could easily define how to meet the existing UR. These were:</p> <ul style="list-style-type: none">- NFR_CODE_READABILITY, to ensure use of white space and indentation- NFR_REFACTOR_LONG_METHODS, as the name suggests

Editing fit criteria for existing non-functional requirements	<p>We changed the following fit criteria for:</p> <ul style="list-style-type: none"> - NFR_PORTABILITY. The criteria was changed from being able to operate on at least 3 different operating systems, to being able to operate on at least 2. The main reason for this was a practical one - our team only has the use of two operating systems - Windows, and a Linux distro - to test the game on. Furthermore, we don't know anyone using MacOS, and so would be unable to test the game on that.
---	--

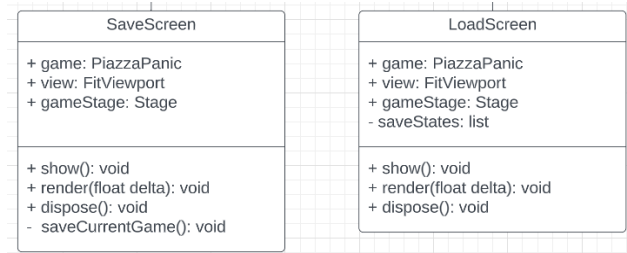
Architecture

Original pdf: <https://imozwastaken.github.io/pdf/Arch1.pdf>

Updated pdf:

Change made	Justification
Extended CRC cards	<p>With new requirements, we had to update the architecture plan to match. For Req18, we added the Powerup card to the existing figure 8</p>  <p>The image shows a CRC card for 'Power Up'. It has a title 'Power Up' and a role 'Cook'. The responsibilities listed are: 'Knows its type', 'Knows position', and 'Structurer'.</p> <p>We didn't make CRC cards for Req16 and Req17, as we decided that these were not accurately represented by CRC cards, and it would be a better use of our time to update the rest of the architecture.</p>
Extended class diagrams	<p>Unlike with the CRC cards, we took the original UML class diagrams as a basis to make entirely new ones (figure 12 in the updated architecture pdf)</p> <p>Req15 was not the child of any other class, and so could exist on its own</p>  <p>The image shows a UML class diagram for 'Powerup'. It has attributes: '+ types:list = [chefSpeedUp, prepareSpeedUp, longerWait, doubleMoney, dontBurn]' and '+ active:Bool'. It has methods: '+ appear(): void' and '+ applyPowerup(Cook): void'.</p> <p>Req16 didn't require another class to be made. Instead, the method setDifficulty() was added to the existing MenuScreen class.</p>  <p>The image shows a UML class diagram for 'MenuScreen'. It has attributes: '+ game: PiazzaPanic', '+ view : FitViewport', and '+ gameStage : Stage'. It has methods: '+ show(): void', '+ render(float delta): void', '+ dispose(): void', and '- setDifficulty(): void'.</p> <p>Req17 required two new main classes - SaveScreen and LoadScreen, both of which were children of the existing Screen interface. These were to handle saving & quitting an existing game, and loading a previously saved game, respectively. LoadScreen had no additional methods, but <i>did</i> have the attribute saveStates,</p>

a list of the currently-saved games. SaveScreen had the additional method saveCurrentGame()



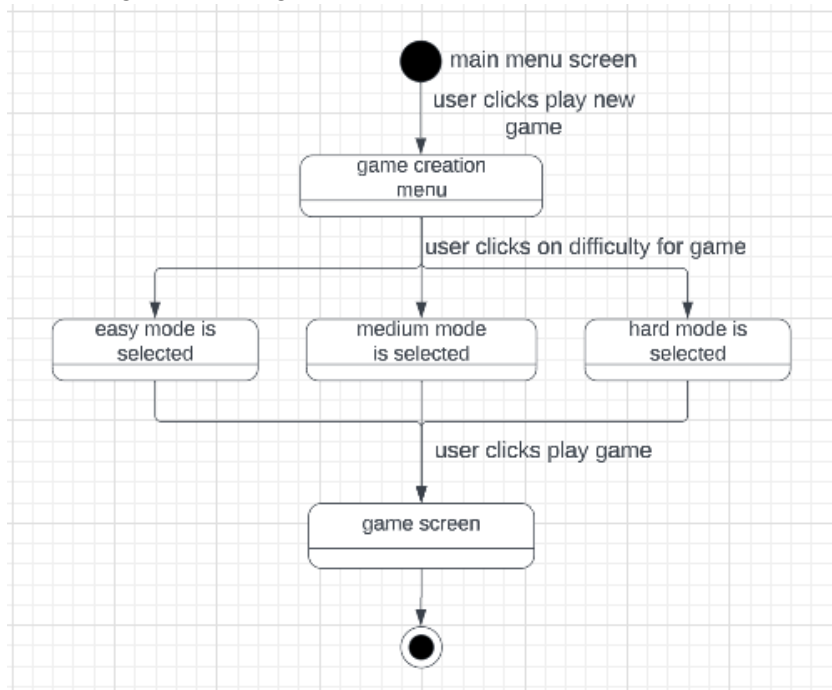
Extended state diagrams

We extended the existing UML state diagrams to represent the new requirements. They demonstrate how the system can change between states, and the events that will trigger said transitions.

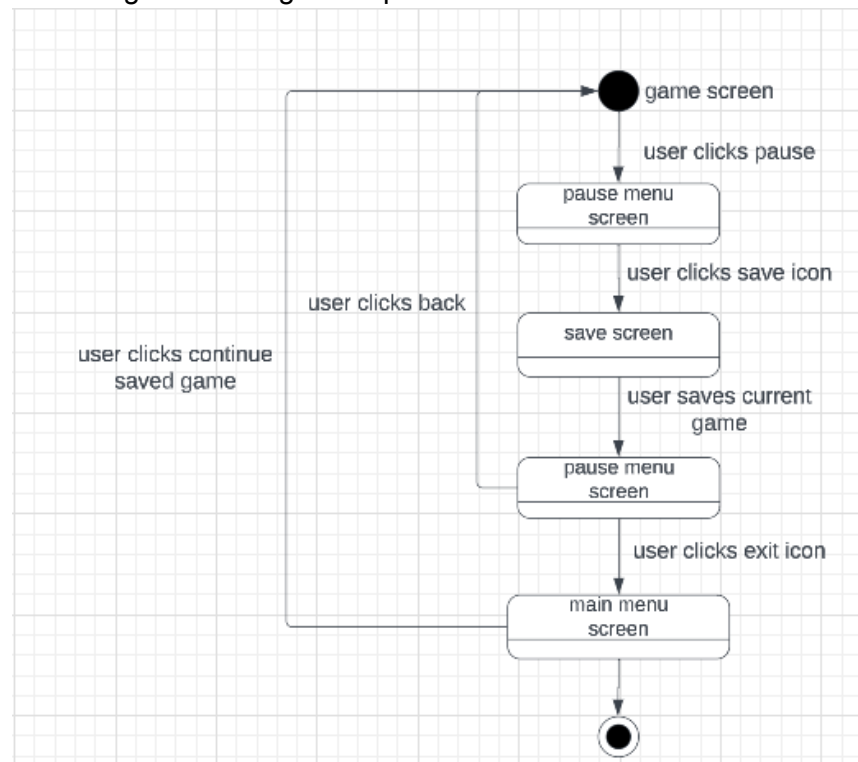
State diagram relating to Req15:



State diagram relating to Req16:



State diagram relating to Req17:



Method Selection + Planning

Original pdf: <https://imozwastaken.github.io/pdf/Plan1.pdf>

Updated pdf:

Change made	Justification
Extending document	<p>We organised our team differently to the original teams, and so detailed this in the table shown in the planning pdf.</p> <p>Furthermore, we created two Gantt charts - one showing the weekly plan, which was put in the planning document, and a more detailed one showing the work done within each week, each update of which was put on the website.</p>
Gantt charts	<p>We created two Gantt charts. One of these was on a weekly level, detailing the work completed by the group as a whole over the week. Our other one showed work done on a more detailed daily level, and also discussed who carried out which work, and showed dependencies between tasks.</p>

Risk Assessment

Original pdf: <https://imozwastaken.github.io/pdf/Risk1.pdf>

Updated pdf:

Change made	Justification
Risk owners	As a new group took over the project, we changed the risk owners to be the new people working on it. In some instances, the risks were identified as no longer relevant in Assessment2, and in this case the owners were Olivia and Nursyarmila working on the change report, as they were deemed to have the greatest overview of what was happening in the project as a whole.
Added risks	As part of our continuous risk assessment, we considered the risks Generic Games had identified, and re-worded and added risks where necessary To ensure backwards compatibility, we did not change any existing risk IDs. The risks we added are: R20 and R24 - relating to testing R21 - relating to continuous integration R22 - relating to current events at uni R23 - detailing further risks from the cloud-based technology we used These are explained and justified in further detail below

R20 + R24

These two risks were identified when considering our need to test the code. R20 discusses the risk of the testing failing on a technical level, for example, due to incorrect code, or if the tests are not thorough enough and thus results in bugs in the code. R24 is a failure on a less technical level - the tests may execute successfully in theory, but the fit criteria chosen are not able to be tested.

The owners of these risks were Muaz and Teddy, the two people in charge of the testing section.

R21

The main risk regarding continuous integration (or rather, the lack of) is that different sections of the code are developed separately, and do not work together.

The owner of this risk was Cameron, in charge of implementing continuous integration, however due to the nature of the risk it also required communication with Zac and Oli, in charge of the implementation side.

R22

This risk was added after the group's experiences with industrial action in late 2022/early 2023. We were conscious of the fact that these strikes, and others planned in the future, could impact the communications between us, the module leaders, and the customer, and so made plans to try and mitigate this impact.

This particular risk had no specific owner, rather, it was something the entire group was bearing in mind throughout.

R23

With our entire project making use of various online technologies to allow us to work collaboratively, a major - though unlikely - risk we identified was the possibility of a given service going down and taking our work with it. Each person was responsible for keeping a backup of their own work in another location, so that it could be recovered if necessary.